Using Multiscale Medial Models to Guide Volume Rendering

David T. Chen, Stephen M. Pizer, J. Turner Whitted

ABSTRACT

We present a hybrid volume rendering method that uses object shape information provided by multiscale medial models to guide a splatting renderer. Such models can efficiently give large-tolerance versions of object boundaries. Our renderer uses the medially implied boundaries to focus the rendering effort towards the region of detailed boundaries, thereby avoiding occlusion of the objects of interest. Our renderer finds a most likely boundary displacement from the medially implied boundary using a Bayesian approach based on the volume directional derivatives of image intensity in directions normal to the medially implied boundary.

CR Categories and Subject Descriptors: I.3.3 [Picture/Image Generation]: *Display algorithms,* I.3.7: [Three-Dimensional Graphics and Realism]: *Color, shading, shadowing, and texture,* I.4.5 [Reconstruction]: *Summation methods,* I.4.6 [Segmentation]: *Pixel classification,* I.4.10 [Image Representation]: *Volumetric.*

Additional Keywords and Phrases: Volume rendering, multiscale medial models, medical visualization, image segmentation.

1 MOTIVATION

Medical imaging technologies such as computed tomography (CT) and magnetic resonance imaging (MRI) have generated a need for effective volume visualization methods, as have techniques such as computational fluid dynamics and finite element analysis. Today volume rendering systems can render

images of large data sets in reasonable amounts of time, anywhere from minutes down to interactive rates depending on image quality and system performance. Parker [15] describes a system that can ray trace isosurfaces in large volumes at interactive rates. Lacroute [9] presents a system that can produce an image of a 256^3 volume in one second without hardware graphics acceleration.

The available techniques for rendering volume data sets fall into two categories – those that extract an iso-intensity surface from the data and continue with conventional surface rendering [13] and those that render the data directly from its samples. Problems with the first approach include misclassification due to reliance on automatic extraction of the surface. Some, but not all [5] practitioners of the latter approach have often made no attempt to classify features in the data, preferring instead to present the data to the user with a minimum of additional processing. Their hope is that the viewer might interpret the displayed data more effectively than a machine. An obvious drawback is that the presentation may end up cluttered with superfluous objects that confuse interpretation as much as machine-introduced artifacts.

For example, in rendering an organ, such as a kidney, within a CT scan of a patient's abdomen, skin, bones or other organs could obscure the kidney. In addition, because regions selected by such iso-intensity methods reflect a weak notion of object, the surfaces extracted have been unstable in situations of low contrast or high image noise.

In this paper we present guided volume rendering, a way to steer the volume rendering process without classification of the volume data, but through the use of an explicit model of the objects which the volumetric data represents. In our example, whereas selecting a density level in the CT volume is adequate to separate bones from the rest of the volume, in regions where the kidney abuts the liver, selecting a density is a poor substitute for selecting a particular soft-tissue organ for visualization.

Steering the rendering process requires at least a gross representation of the region occupied by the object's boundary, i.e., a representation of the shape, size, and pose of the specific object in the image volume. Modern techniques of deformable models may efficiently provide such a representation [3], [8], [23].

Multiscale medial models (MMMs) can be deformed into image data using radius proportional apertures to measure boundary properties at the ends of the basic medial primitives. The multiscale nature of these models provides an explicit tolerance for each boundary position it implies. Therefore an MMM

elegantly handles noisy regions of a data set by treating the noise as small-scale detail that interferes little with the models of larger figures.

Having deformed an MMM into a volume to be rendered and thus having formed an MMM specialized to that volume, the region of specified tolerance about the medially implied boundary [18] can be identified as the only sub-volume interesting to the user. The remainder of the volume can be omitted, thereby removing occluding objects from the rendering and possibly reducing the amount of computation. Moreover, we show below how the medially implied boundary and its tolerance can be used as prior information in determining displacements from the medially implied boundary to the boundary to be rendered.

The visualization method we propose would operate as follows. The user would select from an atlas a generic model of the organ or organs to be rendered. On the gray level slices he might point to a few landmarks associated with the generic model, thus determining an initial placement, orientation, and sizing of the generic model. Then automatically in a few seconds the model would be deformed to the intensity data. At this stage the user could interactively and dynamically vary viewing parameters such as pose and illumination, with the rendering proceeding in real time.

In this paper we cover the process of determining displacements and rendering the displaced boundary by splatting. We assume there exists a process for selecting and deforming the MMM to the specified volume.

2 BACKGROUND

2.1 Splatting

Our method uses a variant of splatting [22] to display boundaries, Splatting treats each voxel in a volume as the result of sampling some continuous volumetric function with a sampling filter. To generate an image, a renderer must take all the samples along with a properly chosen reconstruction kernel to reconstruct the continuous volume function. Because of several nice mathematical properties, Westover chose the Gaussian to be the kernel..

To create a splatting kernel, or footprint, for a voxel, a 3-d Gaussian is transformed from the volume's coordinate system to the viewing coordinate system. This 3-d function is then integrated along the viewing direction and cropped to create 2-d bounded kernel.

To generate an image, each voxel's kernel is blended into the image plane. The voxels are traversed back-to-front to composite them in the proper order. Since the volume is regularly gridded, the correct order of voxels can be specified without sorting them.

To perform proper reconstruction, splatting uses a sheet buffer, a screen-aligned buffer of a certain thickness, and an accumulation buffer. Each pixel in the sheet buffer stores a color and an opacity value, as does the accumulation buffer. The sheet buffer starts at the back of the volume. Every voxel that falls within the sheet buffer's Z thickness is summed into the buffer. Once all the appropriate voxels have been added in, the sheet as a whole is composited into the accumulation buffer. Then the sheet is cleared and advanced forward. Using the sheet buffer produces a more accurate reconstruction than compositing each voxel directly into the accumulation buffer [22].

2.2 Multiscale Medial Models and Their Deformation

Our work uses the object shape, pose, and size described by multiscale medial models (MMMs) [17], [19], [21] as a basis for generating images of volume data sets because of the following apparent properties. It is insensitive to accurate initial placement, efficient, stable against image disturbances, and most especially it provides an explicit tolerance for each boundary position it implies. Alternative models of shape [3], [8], [23] localize and represent object boundaries in one step at the fine resolution needed for rendering. The method we are proposing, in contrast, gains efficiency and stability by first representing the boundaries smoothly and with large tolerance and then refining the surface by computing fine displacements.

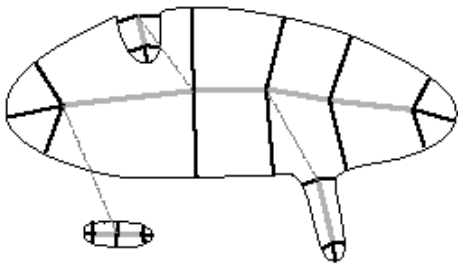
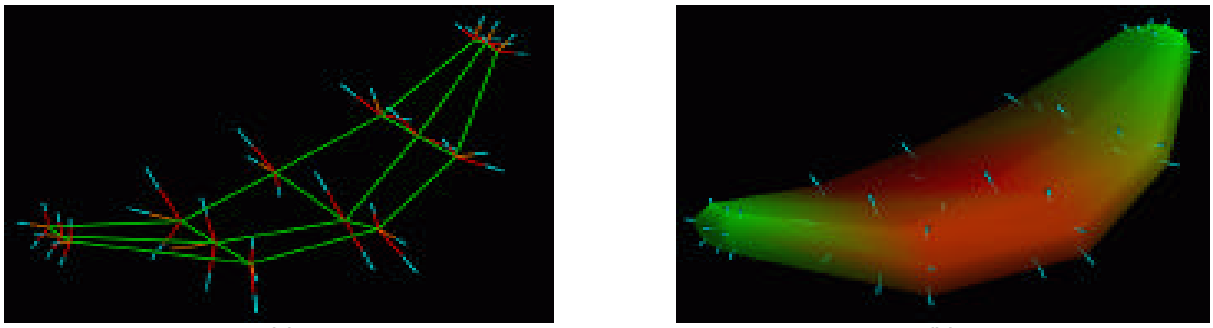


Figure 1. A 2-d multiscale medial model. Medial primitives are linked down the middle of an object. Each medial primitive is also connected to several boundary points. Sub-figures are linked to the main figure in a hierarchy.

Multiscale medial models represent the shape and structure of objects via linked nets of medial primitives, with the primitives typically spaced at about the local width of the object. Figure 1 contains a 2-d example. Each medial primitive has a position on an axial manifold down the middle of the object and two arrows of equal length, being the local radius of the implied object. The object boundary implied by a primitive is orthogonal to each arrow at its endpoint and has a tolerance proportional to the radius. A simple object is represented by a single connected set of such medial primitives. The medial manifolds are usually coarsely sampled – typically a 3-d medial surface might consist of some tens of samples [21]. More complex objects are represented by a hierarchy of such medial manifolds, with the larger scale models representing the major object components, and the smaller scale models representing sub-figures such as protrusions or indentations.

Figure 2 illustrates a simple MMM in 3-d. Figure 2(a) shows a mesh of medial primitives in green. Each medial primitive internal to the mesh consists of a position and two red vectors of equal length originating from the position. The length indicates the local half-width of the implied object. The cyan segments at the ends of the red vectors show the size of the tolerance. Figure 2(b) shows how such a primitive implies two sections of boundary respectively orthogonal to each vector end and with tolerance along the arrow proportional to the vector length. The primitive can be specified by its center position, heading (bisecting the vectors), object angle (giving the orientation of the vectors relative to the heading), and vector length (figural half-width). The heading is approximately in the local plane of the mesh, and the vectors are typically nearly orthogonal to that plane.

Interpolation of the position, width, heading, and object angle along the mesh yields a continuously varying primitive value along a medial surface. Thus the mesh of medial primitives implies a smoothly varying boundary with tolerance (Figure 2(b)), the medially implied boundary (MIB). We call the region within the tolerance bounds the *collar* of the medially implied boundary.



(a) (b)

Figure 2. A 3-d multiscale medial model. (a) the mesh of medial primitives with vectors showing connections to the boundary and the tolerance. (b) the medially implied boundary (MIB).

External elements of the medial manifold in 2-d are the end points. In 3-d the external elements form a border of the medial mesh. Each external primitive involves three boundary-pointing vectors, the two as with internal primitives and the heading vector in addition. The medially implied boundary closes around the medial mesh at such end primitives, forming a maximum of curvature in 2-d and a crest in 3-d.

In 3-d special primitive types exist for the degenerate medial cases, tubular regions (those with circular cross-sections) and spherical objects.

Within a mesh the medial primitives are already linked. Meshes defining protrusions, indentations, enclosed objects, or adjacent objects are linked with their parents by interfigural links, forming a directed, acyclic graph of figures. All of these links indicate relative position, width, heading, and object angle. The protrusion and indentation subfigures modify the boundary and collar implied by the parent figure, breaking out parts of it and replacing it with boundary components and collars defined by the subfigure.

Prototype models in an atlas will have been extracted by a process involving user interaction and image analysis to fit the models to training images. These models may contain statistical information on the shape variability of the models and on the intensity variations across the implied boundary at various locations along that boundary.

Deformation of a model into a new image proceeds after initialization by an iterative coarse-to-fine optimization of a posterior probability, where the prior measures the probability of a particular object deformation and the likelihood function measures the consistency of the image information with the deformed model.

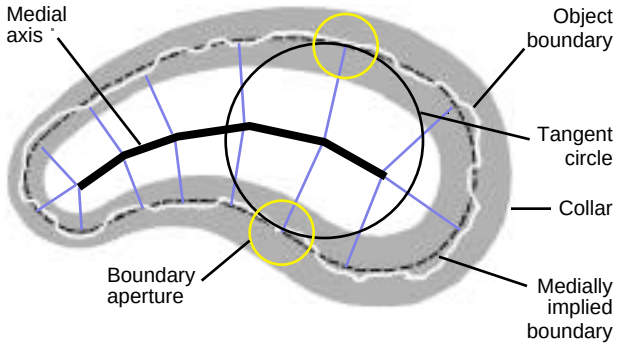


Figure 3. The elements of a multiscale medial model. The object's boundary is measured with an aperture, and opposing boundary points are linked at a medial point. The collar is the region where the boundary can be located.

Object boundaries' contributions to medial information in an MMM are measured by integrating volume directional derivatives over regions of the volume. Finite measurement apertures that are proportional to the object widths (Figure 3) select these regions. Thus the tolerance of a medial primitive describes both its width at a point on the medial surface and the local resolution of the boundary measurement (multiplied by a constant).

MMMs can efficiently locate object boundaries with tolerance in 2-d and 3-d images. They are characterized by stability against noise, against blurring and against interference from neighboring objects. This stability means that a given object is likely to produce the same MMM regardless of noise or blurring [14]. Fritsch [6] and Liu [12] apply MMMs to the problems of 2-d image and 3-d volume registration, respectively. We use MMMs to locate the boundaries of objects for visualization [18].

3 CONTRIBUTION

We have developed a hybrid volume visualization method that uses object shape represented by MMMs to guide a splatting volume renderer. We answer the question "how can volume visualization be improved by steering and how well can we guide the rendering given the object information provided by a medial model?"

We use an MMM to displace the medially implied boundary towards the actual object boundary, contained within the collar. We restrict the regions of the volume that the renderer considers based on MMMs extracted from the objects, via interaction [20] or via organ models [7]. Thus we provide the user with an easy to use method for visualizing only objects of interest and for omitting irrelevant or occluding objects.

We determine an object's boundary displacement by posterior maximization in which prior information is provided by a MMM. The model, specialized to the object in this particular volume by medial strength (sum of all large scale boundary strength measurements associated with a medial primitive), gives information about an object's shape. Using a conditional probability analysis, we demonstrate how to determine the most likely boundary point given the medial model and the image boundary strength at small scale. Finding a boundary as a displacement from the medially implied boundary specifies the surface to be rendered.

Given a collection of sample points on the object boundary, we show how to reconstruct them using a variant of the splatting method. Because our rendering method produces irregularly spaced boundary samples, we adapt splatting to reconstruct an image from such samples by renormalizing the sum of the splats [11].

4 RENDERING AN OBJECT

To produce an image of an object's boundary, a renderer needs to sample that boundary at a rate proportional to the spacing of pixels in the output image. A traditional volume renderer generates an image by resampling the regularly spaced voxels into the regularly spaced pixels. The renderer emphasizes boundaries in the volume by modulating voxel opacities inside or near object boundaries.

To render from a multiscale medial model, the logical starting point is the boundary provided by the model, the medially implied boundary. It is easy to generate samples on the MIB, but to display the actual object boundary, each sample must be displaced towards that boundary, i.e., a boundary sample must be found within each MIB sample's collar. To find a location of a boundary in a volume, previous rendering methods have used the gradient strength of the volume image. The prior information given by the MMM, specifically, the MIB's position and orientation, can supplement the volume's gradient strength information. Using a Bayesian derivation for most probable boundary point, described in sub-section 4.2.1, the renderer displaces each MIB point towards the boundary along the MIB normal.

Because the MIB is only an approximation to the boundary, the displacement of MIB samples towards boundary samples can make image reconstruction more complicated. While we can produce MIB samples at regular pixel intervals, the displaced boundary samples are jittered from the pixel grid.

Applying a traditional reconstruction method to these jittered samples would produce artifacts. We produce a proper image from such jittered samples by adapting Westover's splatting method [22].

The basic model-guided rendering process has three steps. Section 4.1 describes the first step, sampling the MIB. The renderer generates sample points on the MIB that are spaced at regular intervals in screen space, i.e., the projected samples are pixels. The second step, described in section 4.2, displaces each MIB point towards a boundary point in the collar. Searching the collar requires a formulation of the most probable boundary point given the MMM's boundary information and the volume's boundary strength information. Section 4.3 discusses the third step, reconstructing the jittered boundary samples into an image using a modified splatting method.

4.1 Sampling the Medially Implied Boundary

The job of an object renderer is to produce sample points on the boundary of the object and convert these samples into screen pixels. The simplest case of converting boundary samples into screen pixels occurs if there is a one-to-one correspondence between samples and pixels. In this case, each boundary sample becomes a screen pixel, (omitting the problem of proper Z ordering). If boundary samples are difficult or expensive to compute, a common compromise is to generate them at a lower rate than the pixel rate and then perform some sort of reconstruction, e.g., linear, cubic or Gaussian. The reconstruction is simpler if the sampling rate of the boundary is proportional to the image pixel rate.

Generating regularly spaced samples on an object's boundary in a volume is difficult. The volume provides no representation of the boundary, so the only way to find points on the boundary is to search the volume, which is how a traditional volume renderer finds boundaries. However, it is easy to generate regularly spaced samples on the MIB, since the MMM provides a concrete representation.

Because of this ease of sampling, the model-guided renderer starts by generating sample points on the MIB. The renderer represents the MIB as a triangle mesh and rasterizes the triangles to produce samples. The renderer interpolates the following variables across each MIB triangle: screen space position, volume space position, volume space surface normal and MIB tolerance width. The screen space position determines the sample's location in the image. The volume space position specifies the collar

center, where the search for the actual boundary in the volume data starts, and the volume space surface normal specifies the direction of the boundary search in the collar.

4.2 Boundary from the Medially Implied Boundary

4.2.1 General Approach

The MMM provides a region, the collar, where an object's boundary can be found. As discussed in the previous section, the renderer generates regularly spaced samples on an object's MIB and computes a normal at each sample point. Given a sample point on the MIB, the renderer must displace it towards the actual boundary, since our goal is to display the actual boundary. To find a MIB point's boundary location, the renderer needs to search the volume along the boundary normal near that MIB point.

Searching for the boundary involves evaluating a boundary strength function within the collar. The collar for a MIB point is a 1-d space centered at the point and normal to the MIB. The renderer samples a boundary strength function in the 1-d search space at regular intervals. Boundary strength, a function of the medial model and the volume image, typically is a scale-space directional derivative of the data.

Associated with the collar of a MIB sample is a probability density that has infinite extent but falls off quickly away from the MIB. This density function is the medial model portion of the boundary strength function and is modeled as a Gaussian that is maximal at the MIB point and decreases with distance from the MIB. This function leads to a quadratic log density maximal at the MIB point. Figure 4 illustrates how we combine the collar's probability with the volume boundary probability to produce a most-probable boundary point. A derivation of the most probable boundary point follows in sub-section 4.2.1.

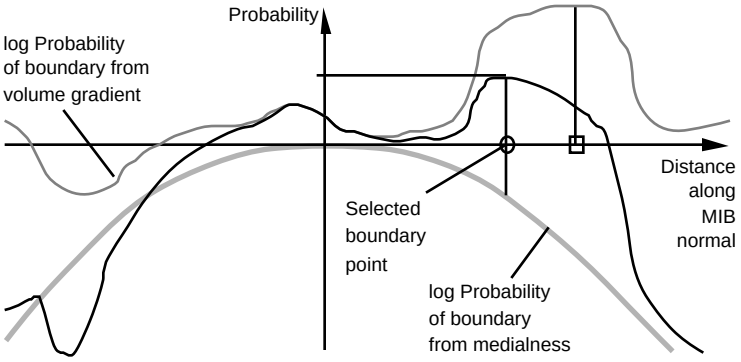


Figure 4. Boundary probability. Two functions, the log of the boundary probability mapped from the volume's directional derivative along the normal and a quadratic centered at the MIB, are added to find the most probable boundary point.

The width of the density function describing the collar is proportional to the MIB's scale, which specifies the resolution to which the boundary's position is known. This scale is a result of the principle that a boundary's resolution as determined by an MMM should be proportional to the object's width. An MIB point's standard deviation is simply the interpolated scale of its medial point multiplied by a factor λ that is provided by the medial model computation method. If a boundary's scale is large, there is more uncertainty about its precise location than if its scale is small.

Within the collar the renderer evaluates a boundary strength function to find a boundary point. The boundary strength along the MIB point's normal determines the volume image portion of the definition of boundary strength. For this we measure the volume gradient dotted with the MIB normal direction, $_{\sigma_B} I \cdot \hat{l}$, which is more appropriate for our method than the more common gradient magnitude, $\| I \|$, because of the medial model's additional boundary direction information. (I is the volume function, σ_B is the MIB scale, proportional to the collar width, and $\hat{l}$ is a unit vector in the MIB normal direction).

If the volume's derivative along the MIB's normal is high, it is more likely that the point is on the boundary. More precisely, we can map the directional derivative into the log probability of being on a boundary. Given a boundary strength formulation, we evaluate it within the collar and add it to the medially implied log probability. The point with the highest probability is chosen as the boundary point and is displayed.

The volume derivative is at the scale σ_B, a constant for a given MIB point. However, different MIB points have different values of σ_B. To make the boundary strength available at the appropriate scale while allowing boundary strength to be calculated as a pre-process, before the boundary strength scale needed at a location is known, the renderer maintains a pyramid containing the volume gradient at all locations and at multiple scales. The gradient pyramid, built in a pre-process, allows fast boundary strength computation at arbitrary scale during the rendering.

The renderer samples the collar at half voxel-width steps and at each step tri-linearly interpolates the volume gradient at the proper scale in the gradient pyramid to compute boundary strength. The step size is ad hoc, but it produces good results. The gradient scale is the MIB scale multiplied by a user-specified constant, and the number of steps along the collar is also proportional to the boundary scale.

Once the renderer has found a boundary point, it shades the point using the Phong lighting model [16]. For the lighting the user has several options for obtaining a surface normal. The primary method is to use the volume gradient taken at a scale based on the MIB's scale and a user-specified constant. The second option is to use the MIB normal. This normal will tend to lose surface detail, since the MIB is found at a large scale (proportional to the object's width), but if the data is noisy will produce a cleaner looking result. Finally the user can blend the gradient and MIB normals. [1] describes how the renderer can blend smoothly between the two, depending upon the amount of local noise detected.

A benefit of our method is that every boundary point is shaded. Traditional volume renderers shade voxels before reconstruction. They assume, for efficiency, that the shading is a band-limited process. If the shading changes faster than the voxel sampling frequency, aliasing artifacts can occur in the shading. Such artifacts are likely in regions where the normals change quickly or where there are small specular highlights. Because our renderer shades every boundary point, such aliasing artifacts do not occur.

4.2.2 Maximum Boundary Probability

A central question of our research is "how can a renderer find a boundary given a volume boundary strength function and a multiscale medial model's boundary information?" Clearly the MMM gives us some knowledge about where the boundary is likely to be found, albeit at a scale typically much larger

than desirable for display. To take advantage of the model's boundary information we have developed a mathematical formulation for the most probable boundary point by applying Wilson's Bayesian formulation [24], [7].

Our method uses the MMM to provide a prior distribution on the locations of object boundary points, i.e., a model of what boundary we expect to see locally. The MIB provides the mean of the prior probability. Combining this prior with the likelihood function derived from the volume leads to the posterior density, whose maximum provides the boundary that is chosen for display.

In this model-based approach, the probability that a given point is a boundary point can be defined as the conditional probability

$$p\left(\vec{x}\middle|B\left(\vec{x},\sigma_B\right), b_M\left(\vec{x}_M,\sigma_M,\vec{l}\right)\right), \tag{1}$$

where $\vec{x}$ is a position within the collar and $B\left(\vec{x},\sigma_B\right)$ is the boundary strength response function of the volume data at the scale of the object boundary, σ_B. $b_M\left(\vec{x}_M,\sigma_M,\vec{l}\right)$ is the medial information concerning the boundary location, which depends upon $\vec{x}_M$, the position of the object middle; σ_M, the scale of the object middle; and $\vec{l}$, the link between the medial point and the MIB point $\vec{x}_B = \vec{x}_M + \vec{l}$. Expression (1) is the probability that a point $\vec{x}$ is a boundary point given the boundary strength response function at $\vec{x}$ at the scale σ_B and the MIB point $\vec{x}_B$. Figure 5 shows the geometric relationships of the symbols.

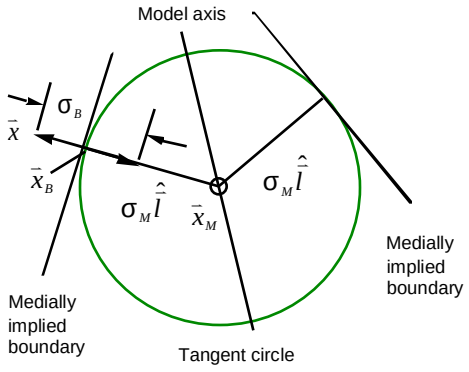


Figure 5. The geometric relationships of MMM variables. σ_B is the scale of the MIB point, which is proportional to $|\vec{l}|$, and $\vec{x}_B$ is the position of the MIB point.

We model the MMM's boundary information as a Gaussian densqity function centered at the MIB. Since this distribution is only a function of the displacement along $\vec{l}$ from the MIB, $b_M\left(\vec{x}_M,\sigma_M,\vec{l}\right)$ can be recast in terms of the random variable *d*, where $\vec{x} = \vec{x}_B + d\,\vec{l}$. Therefore expression (1) can be simplified to $p\left(d|B,\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right)$.

To find the most probable boundary position, the probability should be maximized over *d*, i.e., we should compute

$$Arg\max_d p\left(d|B,\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right).$$

Applying Bayes's theorem for conditional probabilities produces

$$Arg\max_d p\left(d|B,\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right) = Arg\max_d \frac{p\left(B|d,\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right)\,p\left(d|\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right)}{p\left(B|\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right)}.$$

$p\left(B|d,\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right)$ can be written as $p\left(B|\vec{x},\vec{l}\right)$, because the boundary strength measure's probability is entirely determined given the position at which it is computed and the direction normal to the boundary. The probability of boundary strength at a given position and in a given direction is some unimodal function of *B*,

$$p\left(B|\vec{x},\vec{l}\right) = e^{f\left(B(\vec{x},\vec{l})\right)} \qquad (2)$$

The function can be given in a lookup table. For mathematical simplicity we used the function $f(B) = kB$ used in [24]. In the range of boundary strength responses seen in our work, this probability works well enough.

A Gaussian distribution represents the medial model's boundary information. That is,

$$p\left(d|\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right) = G\left(d;\beta\sigma_M\right),$$

a zero-mean Gaussian with standard deviation $\beta\sigma_M$, so

$$\log p\left(d|\left\{\vec{x}_B,\vec{l},\sigma_M\right\}\right) = c - d^2/2\left(\beta\sigma_M\right)^2 \qquad (3)$$

β is a user-specified constant, and *c* is another constant.

Maximizing the probability is the same as maximizing the logarithm of the probability. Therefore we compute

$$Arg\max_d \left[\log\left(p\left(B \middle| \vec{x}, \vec{l} \right) \right) + \log\left(p\left(d \middle| \vec{x}_B, \vec{l}, \sigma_M \right) \right) - \log\left(p\left(B \middle| \vec{x}_B, \vec{l}, \sigma_M \right) \right) \right].$$

Since the last term of the above expression is not dependent on *d*, this result is the same as

$$Arg\max_d \left[\log\left(p\left(B \middle| \vec{x}, \vec{l} \right) \right) + \log\left(p\left(d \middle| \vec{x}_B, \vec{l}, \sigma_M \right) \right) \right] \tag{4}$$

Substituting in equations (2) and (3) into (4) produces

$$d_{opt} = Arg\max_d \left[kB\left(\vec{x}, \vec{l} \right) - d^2 / 2\left(\beta\sigma_M \right)^2 \right] \tag{5}$$

The constant c from equation (3) is not dependent on *d* and has been dropped.

The boundary strength response function is the directional derivative of the volume function I along the $\vec{l}$ direction. Furthermore, this function is integrated over a region whose size is proportional to the scale of the boundary. Thus the boundary strength response function is expressed as ${}_{\sigma_B} I\ \hat{\vec{l}}$ ("^" indicating normalization to unit length). Inserting this boundary strength function into equation (5) yields the final expression for the displacement *d* along $\vec{l}$,

$$Arg\max_d \left[k\ {}_{\sigma_B} I\ \hat{\vec{l}} - d^2 / 2\left(\beta\sigma_B \right)^2 \right]$$

Earlier figure 4 showed an example of the three probability functions: the logarithm of a boundary strength response function, the logarithm of a medial model probability function and the sum of the two. Note how the medial prior biases the boundary towards the MIB.

Figure 6 shows the result of the model-guided renderer using a MRI scan of a gourd. Figure 6(a) shows the gourd's MIB. Figure 6(b) is an image produced by the renderer. Clearly there is more surface detail in the rendered image.

(a) (b)

Figure 6. Rendering a gourd. (a) the MIB of a MRI scan of a gourd. (b) an image of the gourd produced by the model-guided renderer. The surface detail present in the rendered image is missing from the MIB.

Figure 7 shows slices from the gourd's volume. The locations of boundary samples produced in the rendering are overlaid on the slices, highlighting the gourd's boundary. Closer inspection of the MRI data and the superimposed boundary reveals that the estimated surface is indeed a more faithful representation of the object boundary. Of course, the fidelity of the MRI data to the actual object's shape is an important question but beyond the scope of our work.

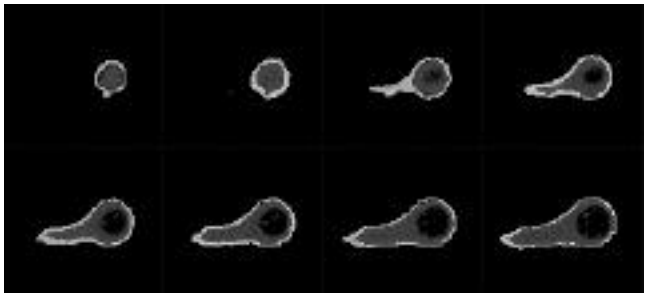
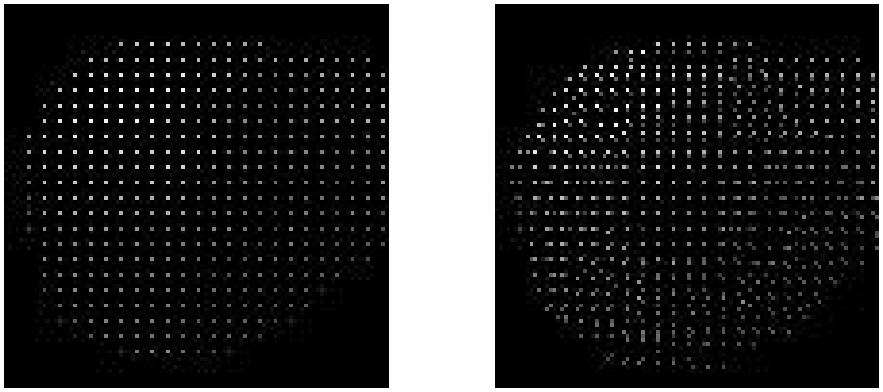


Figure 7. The boundary in the volume. This series of images are slices of the MRI scanned volume of the gourd. The locations of the boundary samples used to generate the rendered image in figure 6 are highlighted in the slices.

Figure 8 shows how going from MIB samples along the medially implied normals to boundary samples produces a displacement in screen space location. Figure 8(a) is a set of regularly spaced samples on the MIB; the sampling rate is coarser than the one used in figure 6. Figure 8(b) shows the samples after the collar has been searched and the sample displaced towards the boundary. The spacing has become irregular. Also, there appear to be more samples because the samples on both the front and backsides of the gourd are visible. In figure 8(a), the front and back samples lie on the sample pixels.



(a) (b)

Figure 8. Screen space images of MIB and boundary locations (a) the locations of MIB samples for a coarsely sampled medial model. (b) the samples after the shift towards the boundary. The regular spacing has been disrupted.

4.3. Reconstructing Jittered Samples

An operation that is implicit in much of computer graphics is reconstructing a continuous function from sampled data and then resampling this function. There has been much work on the instance of the problem where the initial sampling rate is constant, but not as much for irregularly spaced samples. Because the model-guided renderer produces image points that are displaced from a regular grid, it requires a method to reconstruct the continuous surface from such jittered or randomly perturbed points.

Much of the work on reconstructing jittered samples is related to anti-aliasing [4] or distributed ray-tracing [10], [2]. In these situations a scene is sampled multiple times to determine the color of a pixel in the display image. Jittering the positions of the samples from a sub-pixel grid reduces aliasing artifacts. In our work typically the scene is undersampled, for efficiency, i.e., there are fewer samples than display pixels, unlike the previous work. Our method of reconstruction is similar to the adaptive filtering described in [4] except applied to our circumstance.

We reconstruct boundary samples using a variant of Westover's splatting [22]. If we simply used Westover's method on irregularly spaced samples, the result would be bright regions where the samples are denser and dim regions where the samples are sparser. However, examining the reconstruction process suggests how splatting can be adapted for jittered sample points.

Reconstruction starts with a collection of samples of some continuous function. The goal of reconstruction is to reproduce that function so it can be resampled at some other rate. In volume

rendering, the reconstructed continuous function is also projected into screen space before resampling at the pixel-sampling rate.

A ray casting volume renderer reconstructs the volume samples into a 3-d function, which is then resampled along the ray and interpolated to produce a continuous function on the ray. At each sample position on the ray tri-linear interpolation produces a weighted average of the values of surrounding voxels. The ray function is then blended to produce an image sample. In traditional splatting, the continuous 3-d volume function is projected directly into image space, and then the projected function is resampled into pixels using Gaussian weights to average the voxels that affect each pixel. In both methods the weights are designed to sum to 1 or less. This weighted average answers the question "Which voxels affect this pixel and by how much?"

Normalizing each pixel by dividing the intensity by the sum of the weights of the voxels that affect it produces a normalized weighted average, which compensates for the jittered sample locations.

To perform this pixel normalization, we augment the sheet buffer to store not only the weighted sum of samples that touch each pixel, $\sum_{\text{all samples}}^{i} w_i \ f_i$, but also the sum of the weights of those samples, $\sum_{\text{all samples}}^{i} w_i$. w_i is the weight given by the splat footprint at the pixel location, and f_i is the sample value. After the renderer fills the sheet buffer, we normalize pixels where the sum of the weights is greater than 1, resulting in $\sum_{\text{all samples}}^{i} w_i \ f_i \Big/ \sum_{\text{all samples}}^{i} w_i$.

Normalizing all pixels could produce hard edge artifacts at pixels that have weights that correctly sum to less than 1. To avoid this problem, the pixels with weights summing to less than 1 are not normalized. At these pixels the function is undersampled, and leaving them unnormalized, i.e., somewhat dimmed, is a reasonable way to reflect the lack of data.

Figure 9 illustrates a simple 1-d example of the reconstruction of jittered sample points. The original function is a constant with intensity 1. The circles on the X axis represent the locations of the samples, and the light lines represent each sample's reconstruction kernel. The goal of reconstruction is to reproduce the original function from the set of samples. The dashed line, representing the result of summing the Gaussian splat kernels, shows a rise in the reconstructed function where the samples are

closely spaced. The dark line represents the normalized reconstructed function. There remains one dip in the normalized function on the right, where the samples are too far apart, but overall it reproduces the original function much more closely than the unnormalized reconstructed function.

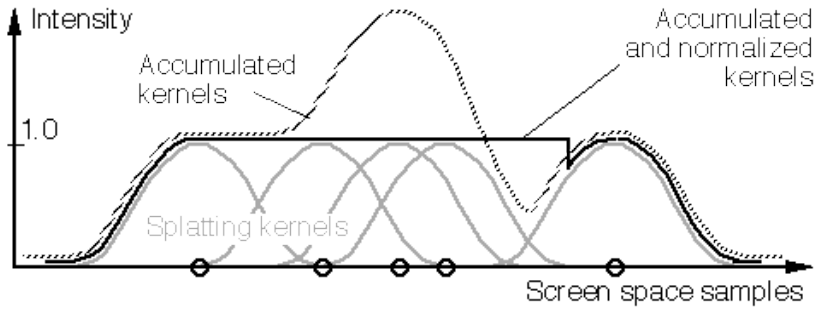
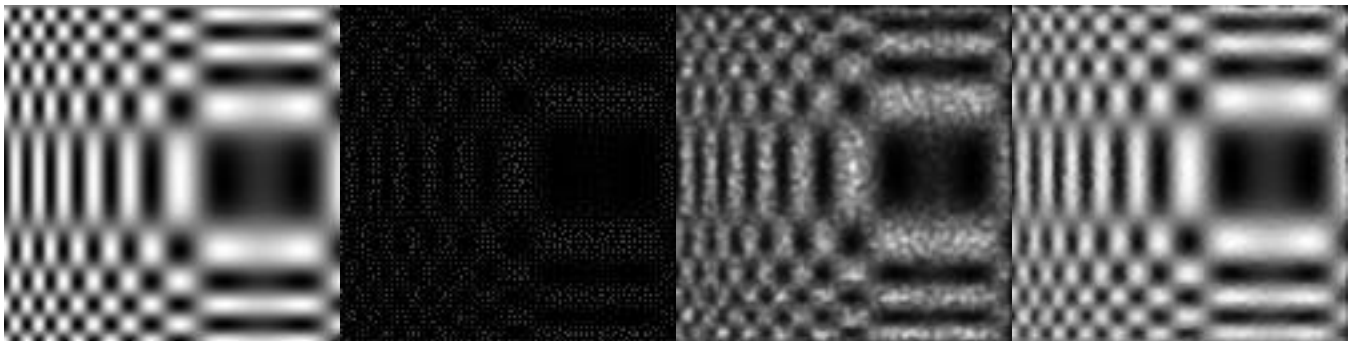


Figure 9. Reconstructing jittered samplesThe accumulated kernels are normalized to reconstruct the original function. Where samples are too far apart, the reconstructed function can dip.

Figure 10 shows some results of the reconstruction method. Traditional and normalized splatting are applied to samples taken from a sinusoidal test image. Figure 10(a) shows the source image, and figure 10(b) shows the locations at which the image was sampled. The sampling rate is 1 in 4 pixels on a jittered grid. Figure 10(c) is the jittered samples reconstructed with traditional splatting. Figure 10(d) shows how normalized splatting reconstructs the samples. Using a slightly larger kernel can smooth out some of the artifacts.



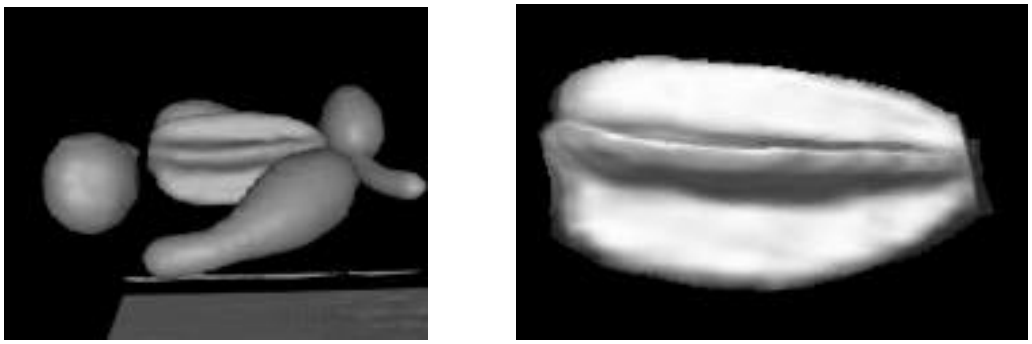
(a) (b) (c) (d)

Figure 10. Jittered splatting. (a) the original test image. (b) the sample locations. (c) a reconstructed image using traditional splatting. (d) a reconstructed image using normalized splatting.

5. RESULTS

Figure 11 shows two renderings of a MRI scan of a starfruit surrounded by four gourds. Figure 11(a) was produced using a ray tracer. Figure 11(b) was produced using our model-guided renderer. These images demonstrate our ability to render only interesting objects. Clearly a traditional volume renderer could render only the starfruit by using cutting planes. However configurations where cutting planes

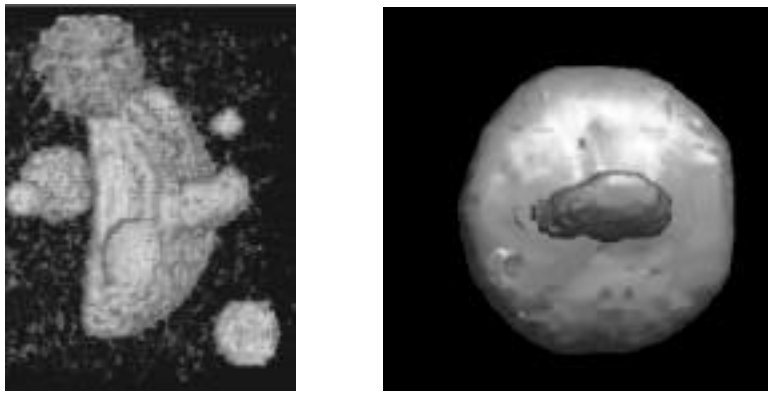
would not work are easy to imagine. The model-based method can select the starfruit regardless of its surroundings.



(a) (b)

Figure 11. Vegetable volume. (a) a ray traced image of a starfruit surrounded by 4 gourds. (b) the starfruit selected by the multiscale medial model-guided renderer.

Figure 12 shows two renderings of a test object subjected to several disturbances. A very high level of random noise has been added, several occluding spheres have been inserted and two sections of spheres have been cut out of the object. Figure 12(a) was rendered with a traditional ray tracer. Figure 12(b) is the same noisy, perturbed volume rendered by the model-guided renderer. The noise and spheres have been removed, and where the holes were, the surface of the object has been interpolated. The details of the medial model based noise reduction and surface interpolation methods can be found in [1].

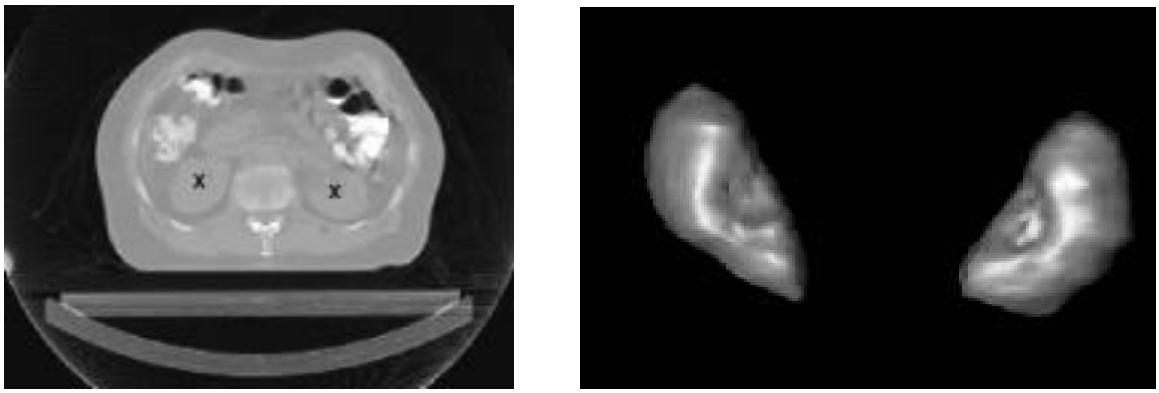


(a) (b)

Figure 12. Noisy test object. (a) a ray traced image of a noisy test object occluded by spheres. (b) the object selected by the model-guided renderer. The object's boundary has been interpolated using the model's shape information to reduce the noise.

Figure 13 shows the model guided renderer applied to a real medical volume. Figure 13(a) is a slice of a CT scan a pair of kidneys, centered at the X's, and the surrounding abdomen. Because the density levels of the kidneys and the surrounding tissue are very similar, a traditional volume renderer

would not be able to separate the kidneys. We were in fact unable to find them with our ray tracer. Figure 13(b) shows the kidneys rendered by the model-guided renderer.



(a) (b)

Figure 13. Kidneys. (a) a slice from a MRI scan of an abdomen. (b) the kidneys rendered by our system.

6. CONCLUSIONS AND FUTURE WORK

Our research presents a new method for using multiscale medial models to guide volume visualization. Traditional volume visualization methods have problems dealing with occlusion, typically selecting regions based on cutting planes or intensity values. Our renderer allows a user to select regions of the volume based on objects, such as specific organs in a patient, a much more intuitive method.

This paper described how to displace a medially implied boundary point towards the actual boundary. We presented a conditional probability analysis to derive the most probable boundary point given the medial model's boundary information and the volume's gradient-based boundary information. The medial model's boundary distribution is modeled using a Gaussian centered at the medially implied boundary with a standard deviation proportional to the object's width. The volume's boundary probability is the derivative of the volume taken in the direction of the MIB's normal, because the model gives an approximate boundary orientation.

Our system could be improved by allowing the user to select the rendering resolution for specific objects or based on an object's scale. In fact any rendering parameter could vary by object, such as opacity ramps or the weighting of image boundary versus medially implied boundary displacement.

Other possible areas of research include extending the range of object model types used to guide the rendering. An active shape model is a boundary based object model that uses global shape to fit a model to an image [3]. A model is allowed to deform in ways predefined by training data. Also, an active

shape model contains a statistical model the expected gray level in regions around each model point. Incorporating such a statistical model into the method could improve the boundary finding procedure.

As for the application of multiscale medial models, various other areas of computer graphics would benefit. The primary feature of these models is that they decompose objects into larger scale global structure with a tolerance and width, and smaller scale detail. The scale space decomposition of structure allows the level of detail to be matched to the level needed for design, display, physically based modeling or any other problem in graphics.

Other applications could take advantage of the multiscale nature of the medial model. Multiscale medial models could be used to direct object simplification; e.g., small-scale detail could be reduced without changing global shape. Multiscale medial models also could be applied to modeling by allowing a user to work on different levels of detail while the model combines the levels into a complete object. The hierarchy of models allows figural homologies between objects. Matching figures leads directly to geometric morphing between objects in addition to the model based localization of objects demonstrated in our work.

7. ACKNOWLEDGEMENTS

We gratefully acknowledge Andrei State and Andrew Thall for help with the writing and illustrations, Daniel Fritsch and Kah-Chan Low for the software for representing and deforming multiscale medial models, and Stephen and Clara Chen for their support. This work was done under the partial support of NIH grant P01 CA47982.

8. REFERENCES

[1] D.T. CHEN, *Volume Rendering Guided by Multiscale Medial Models*, Ph. D. dissertation, Dept. of Computer Science, Univ. of North Carolina, 1998.

[2] R.L. COOK, "Stochastic Sampling in Computer Graphics," *ACM Transactions on Graphics*, **5**(1):51-72, Jan. 1986.

[3] T.F. COOTES, A. HILL, C.J. TAYLOR, and J. HASLAM, "The Use of Active Shape Models for Locating Structures in Medical Images," *Image and Vision Computing*, **12**(6):355-366, July 94.

[4] M.A.Z. DIPPE and E.H. WOLD, "Antialiasing Through Stochastic Sampling," *Computer Graphics*, **19**(3):69-78, July 1985.

[5] R.A. DREBIN, L. CARPENTER, and P. HANRAHAN, "Volume Rendering," *Computer Graphics*, **22**(4):65-74, Aug. 1988.

[6] D.S. FRITSCH, S.M. PIZER, B.S. MORSE, D.H. EBERLY, and A. LIU, "The multiscale medial axis and its applications in image registration," *Pattern Recognition Letters*, **15**:445-452, 1994.

[7] D.S. FRITSCH, S.M. PIZER, L. YU, V.E. JOHNSON, and E.L. CHANEY, "Localization and Segmentation of Medical Image Objects using Deformable Shape Loci," *IPMI*, 1997.

[8] A. KELEMEN, G. SZEKELY, and G. GERIG, "Three-dimensional model-based segmentation," *Proc. Workshop on Biomedical Image Analysis*, IEEE Computer Science Press: 4-13, 1998.

[9] P. LACROUTE and M. LEVOY, "Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transform," *Computer Graphics*, **28**(4):451-458, July 1994.

[10] M.E. LEE, R.A. REDNER, and S.P. USELTON, "Statistically Optimized Sampling for Distributed Ray Tracing," *Computer Graphics*, **19**(3):61-68, July 1985.

[11] M. LEVOY and J.T. WHITTED, "The Use of Points as a Display Primitive," *UNC Computer Science technical report*, TR85-022, 1985.

[12] A. LIU, S.M. PIZER, D.H. EBERLY, B.S. MORSE, J. ROSENMAN, E.L. CHANEY, E. BULLITT, and V.E. CARRASCO, "Volume registration using the 3D core," *VBC '94*, SPIE, **2659**:217-226, 1994.

[13] W. LORENSEN and H. CLINE, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *Computer Graphics*, **21**(4):163-170, July 1987.

[14] B.S. MORSE, S.M. PIZER, D.T. PUFF, and C. GU, "Zoom-Invariant Vision of Figural Shape: Effects of Cores on Image Disturbances," to appear *Computer Vision and Image Understanding* 1998.

[15] S. PARKER, P. SHIRLEY, Y. LIVNAT, C. HANSEN, and P. SLOAN, "Interactive Ray Tracing for Isosurface Rendering," *Proc. Of IEEE Visualization '98*, 233-238, 1998.

[16] B.T. PHONG, "Illumination for Computer Generated Images," *Communications of the ACM*, **18**(6):311-317, June 1975.

[17] S.M. PIZER, C.A. BURBECK, J.M. COGGINS, D.S. FRITSCH, and B.S. MORSE, "Object shape before boundary shape: scale-space medial axes," presented at Shape in Picture, (NATO Advanced Research Workshop), Journal of Mathematical Imaging and Vision, **4**:303-313, 1994.

[18] S.M. PIZER, S. MURTHY, and D.T. CHEN, "Core-based Boundary Claiming," *Proc. SPIE Medical Imaging 94: Image Processing*, **2167**, 1994.

[19] S.M. PIZER, D. EBERLY, B.S. MORSE, and D.S. FRITSCH, "Zoom-Invariant Vision of Figural Shape: The Mathematics of Cores," *Computer Vision and Image Understanding*, **69**:55-71, 1998.

[20] S.M. PIZER, D.S. FRITSCH, K.C. LOW, and J.D. FURST, "2D & 3D figural models of anatomic objects from medical images," *Mathematical Morphology and Its Applications to Image Processing*, HJAM Heijmans, JBTM Roerdink, eds. (invited paper, Proc. ISMM '98), Kluwer Computational Imaging and Vision Series: 139-150, 1998.

[21] S.M. PIZER, D.S. FRITCH, P.A. YUSHKEVICH, V.E. JOHNSON, and E.L. CHANEY, "Segmentation, Registration, and Measurement of Shape Variation via Image Object Shape", *UNC Computer Science technical report* submitted for publication, 1999.

[22] L. WESTOVER, *Splatting - A Parallel, Feed-Forward Volume Rendering Algorithm*, Ph. D. dissertation, Dept. of Computer Science, Univ. of North Carolina, TR91-029, 1991.

[23] R.T. WHITAKER and D.T. CHEN, "Embedded Active Surfaces for Volume Visualization," *Proc. SPIE Medical Imaging 94: Image Processing*, **2167**, 1994.

[24] A. WILSON, V.E. JOHNSON, S.M. PIZER, D.S. FRITSCH, L. YU, and E.L. CHANEY, "Towards a Framework for Automated Image Analysis," *Proc. 16th Leeds Annual Statistical Workshop: Image Fusion and Shape Variability Techniques*, Univ. of Leeds Press, Leeds, England, 1996.